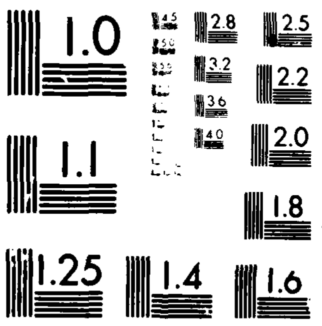
AD-A085 190 MASSACHUSETTS INST OF TECH CAMBRIDGE OPERATIONS RESE--ETC F/G 12/2
DISAGGREGATION AND RESOURCE ALLOCATION USING CONVEX KNAPSACK PR--ETC(U)
MAR 80 G R BITRAN, A C HAX N00014-75-C-0556
UNCLASSIFIED TR-175 NL

1 OF 1
AD
A085190

END
DATE
FILMED
6-80
DTIC


1.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

2

DISAGGREGATION AND RESOURCE ALLOCATION USING CONVEX KNAPSACK PROBLEMS WITH BOUNDED VARIABLES

by

GABRIEL R. BITRAN

and

ARNOLDO C. HAX

Technical Report No. 175

OPERATIONS RESEARCH CENTER

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

March 1980

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report No. 175	2. GOVT ACCESSION NO. AD-A085 190	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) DISAGGREGATION AND RESOURCE ALLOCATION USING CONVEX KNAPSACK PROBLEMS WITH BOUNDED VARIABLES,		5. ~~TYPE OF REPORT &~~ PERIOD COVERED (9) Technical Rep[illegible]t, ~~March 1980~~
		6. PERFORMING ORG. REPORT NUMBER
7. ~~AUTHOR(s)~~ (10) Gabriel R./Bitran Arnoldo C./Hax		8. ~~CONTRACT OR GRANT~~ NUMBER(s) (15) NØØØ14-75-C-Ø556
9. PERFORMING ORGANIZATION NAME AND ADDRESS M.I.T. Operations Research Center 77 Massachusetts Avenue Cambridge, Massachusetts 02139 (12) 23		10. ~~PROGRAM ELEMENT, PROJECT,~~ TASK AREA & WORK UNIT NUMBERS NR 347-027
11. CONTROLLING OFFICE NAME AND ADDRESS O.R. Branch, ONR Navy Department 800 North Quincy Street Arlington, VA 22217 (11)		12. ~~REPORT DATE~~ Mar[redacted]8Ø
		13. ~~NUMBER OF PAGES~~ 19 pages
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Releasable without limitation on dissemination.

(14) TR-175

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Resource Allocation
Aggregate Production Planning
Production Planning
Disaggregation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

See page ii.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified 270720 [illegible]

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

DISAGGREGATION AND RESOURCE ALLOCATION USING CONVEX KNAPSACK PROBLEMS WITH BOUNDED VARIABLES

by

GABRIEL R. BITRAN

and

ARNOLDO C. HAX

Technical Report No. 175

Work Performed Under

Contract N00014-75-C-0556, Office of Naval Research

Multilevel Logistics Organization Models

Project No. NR 347-027

Operations Research Center

Massachusetts Institute of Technology

Cambridge, Massachusetts 02139

March 1980

FOREWORD

The Operations Research Center at the Massachusetts Institute of Technology is an interdepartmental activity devoted to graduate education and research in the field of operations research. The work of the Center is supported, in part, by government contracts and grants. The work reported herein was supported by the Office of Naval Research under Contract No. N00014-75-C-0556.

Richard C. Larson
Jeremy F. Shapiro

Co-Directors

ABSTRACT

Disaggregation and resource allocation problems can be often formulated as convex knapsack problems with bounded variables. In this paper, we provide a recursive procedure to solve such problems. The method differs from classical optimization algorithms of convex programming in that it determines at each iteration the optimal value of at least one variable. Applications and computational results are presented.

1. INTRODUCTION

In this paper we present a recursive method to solve the following continuous knapsack problem with bounded variables:

$$(N): \qquad z = \min \sum_{j\varepsilon J^\circ} G_j(x_j)$$

$$\sum_{j\varepsilon J^\circ} x_j = P^\circ \qquad (1)$$

$$\ell b_j \le x_j \le ub_j \qquad j\varepsilon J^\circ \qquad (2)$$

$$x_j \varepsilon X_j \qquad j\varepsilon J^\circ$$

where $G_j(\cdot)$ for $j\varepsilon J^\circ$ are differentiable convex functions on the open convex sets $X_j \subseteq R$, $j\varepsilon J^\circ$ respectively.

Problem (N) arises in many different settings. We briefly illustrate some of the major applications.

In production planning and scheduling, problem (N) arises in the following way. The allocation of production resources, as for example labor hours or machine capacity, is made with aggregate data. The results are aggregate plans that must be implemented at the detailed level and therefore need to be disaggregated. It is at this point that problem (N) plays an important role. Two of the objective functions encountered in these situations are

a) $G_j(x_j) = \dfrac{S_j D_j}{x_j} + \dfrac{1}{2} h_j x_j \qquad j\varepsilon J^\circ$

where for each item j, S_j, D_j, h_j, and x_j correspond to the setup cost, the demand, the holding cost per period and the number of hours of labor to be assigned to the item.

b) $$G_j(x_j) = \left(\frac{P^\circ - \sum_{k\varepsilon J^\circ} (AI_k - SS_k)}{\sum_{k\varepsilon J^\circ} D_k} - \frac{x_j - AI_j + SS_j}{D_j} \right)^2 \qquad j\varepsilon J^\circ$$

The quantity P° represents the total number of labor hours assigned by the aggregate plan to the family composed by the set of items $\{k: k\varepsilon J^\circ\}$. For any generic item $k\varepsilon J^\circ$, AI_k, SS_k, D_k, and x_k are respectively, the available inventory, the safety stock, the demand, and the number of labor hours to be allocated to that item. In this case, problem (N) attempts to equalize the run out time of each item j, i.e., the length of time that the number of units produced during x_j will last, with the run out time of the family $\{k: k\varepsilon J^\circ\}$.

Constraint (1) typically assures that the disaggregation process is consistent with the aggregate plan while, the lower and upper bounds (2) guarantee respectively that the demand for each item will be met and that the overstock limit will not be exceeded. Other applications to production planning can be found in [2] and [3]. A related application is the extensive use of problem (N) in inventory control to dtermine lot sizes and buffer stocks under aggregate constraints (see [6] and [8]).

Very often the allocation of financial resources gives rise to knapsack problems with bounded variables. A typical objective function in this instance is

$$G_j(x_j) = -s_j(x_j+c_j)/(x_j+m_j) \quad \text{with } m_j > c_j \qquad j\varepsilon J^\circ$$

$-G_j(x_j)$ represents the return of investing x_j units in activity j. Constraint (1) indicates that the total amount available for investment must equal P° and the lower and upper bounds are imposed by the market. Of course, the function $G_j(x_j)$ need not be identical for all $j\varepsilon J^\circ$.

Some other contexts where problem (N) has been applied are the theory of search [4], the allocation of promotional resources among competing activities [9], [11], and subgradient optimization [5].

In many instances where (N) plays an important role the set $J°$ may have several hundreds or thousands of elements. Moreover, the problem has to be solved many times. In hierarchical production planning, for example, (see [2] and [3]), problem (N) typically is used in two or more levels of disaggregation and is solved at the beginning of each period. Consequently, an efficient method to solve the continuous knapsack problem with bounded variables is of central interest to many applications.

Formerly, this problem has been treated, for particular objective functions, by convex programming arguments [4] and by dynamic programming [11]. Luss and Gupta [9] have presented an iterative method mainly for strictly convex decreasing functions and a one pass algorithm for a set of particular functions with the variables bounded from below. Luss and Gupta's method consists in relaxing the upper bounding constraints. The algorithm proposed in this paper applies for a more general set of functions whenever the simpler problem of the form min $\{\Sigma\ G_j(x_j) \cdot \text{st}\ \Sigma x_j = P,\ x_j \varepsilon X_j\}$, obtained by relaxing both bounds (2) and solved at each iteration, has a solution. This condition is met in most practical applications. We also provide results for the case where the relaxed problem has no solution. At the end of each iteration of our algorithm (except possibly the last),we show that either the subset of variables $\{x_j : x_j \geq ub_j\}$ have optimal value ub_j in (N) or the subset $\{x_j : x_j \leq \ell b_j\}$ have optimal value ℓb_j in (N). This characteristic of the algorithm differs from classical convex programming methods. More recently Armstrong, Cook, and Palacios-Gomez [1] developed a branch and bound algorithm to solve a problem related to (N). They require the variables x_j to be either equal to zero or to be in the interval $[1,u_j] j\varepsilon J°$ and the functions $G_j(\ )\ j\varepsilon J°$ are assumed to be concave increasing on $[1,u_j]$ and zero elsewhere.

2. THE ALGORITHM

In section 1, we assumed that the functions $G_j(x_j)$ are convex and differentiable. Without loss of generality we can add the conditions:

a) $ub_j > \ell b_j$ $j\varepsilon J°$ since, if $ub_k = \ell b_k$ for some $k\varepsilon J°$ the value of x_k is determined. Hence, k can be deleted from J° and P° replaced by $P°-x_k$.

b) $\sum_{j\varepsilon J°} \ell b_j < P° < \sum_{j\varepsilon J°} ub_j$. Otherwise the problem is either infeasible or the solution is trivially determined.

(N) is a convex problem with linear constraints. Hence, the corresponding Kuhn-Tucker conditions (3) - (9), below, are necessary and sufficient [10] for optimality of x_j $j\varepsilon J°$.

Let $DG_j(x_j)$ denote the derivative of $G_j(\cdot)$ at the point x_j and let x_j^* for $j\varepsilon J°$ be an optimal solution to (N). The Kuhn-Tucker conditions are:

$$DG_j(x_j^*) + \lambda + u_j - \tau_j = 0 \qquad j\varepsilon J° \qquad (3)$$

$$u_j(x_j^*-ub_j) = 0 \qquad j\varepsilon J° \qquad (4)$$

$$\tau_j(\ell b_j-x_j^*) = 0 \qquad j\varepsilon J° \qquad (5)$$

$$\sum_{j\varepsilon J°} x_j^* = P° \qquad (6)$$

$$\ell b_j \leq x_j^* \leq ub_j \qquad j\varepsilon J° \qquad (7)$$

$$\lambda\varepsilon R,\ u_j \geq 0,\ \tau_j \geq 0 \qquad j\varepsilon J° \qquad (8)$$

$$x_j^* \varepsilon X_j \qquad j\varepsilon J° \ . \qquad (9)$$

λ, u_j, and τ_j are the Kuhn-Tucker multipliers associated with the constraints (1), $ub_j-x_j \geq 0$ and $x_j-\ell b_j \geq 0$ $j\varepsilon J°$ respectively. When $ub_j = +\infty$ [$\ell b_j = -\infty$] for some $j\varepsilon J°$, the corresponding condition (4) [(5)] and multiplier u_j [τ_j] are deleted.

We first state and prove the optimality of the algorithm under the assumption that all relaxations N(β) of (N) encountered in step 1 have an optimal solution. At the end of this section we relax this assumption.

The Algorithm

Initialization: $J^1 = J^\circ$, $P^1 = P^\circ$.

Iteration β (β=1,2,3...)

Step 1: Solve $N(\beta) : \min \{ \sum_{j\varepsilon J^\beta} G_j(x_j) : \sum_{j\varepsilon J^\beta} x_j = P^\beta, x_j \varepsilon X_j \; j\varepsilon J^\beta\}$

and let the solution be $x_j^\beta \; j\varepsilon J^\beta$. If $\ell b_j \leq x_j^\beta \leq ub_j \; j\varepsilon J^\beta$ define $x_j^* = x_j^\beta \; j\varepsilon J^\beta$ and stop, the solution $x_j^* \; j\varepsilon J^\circ$ generated by the algorithm is optimal. Otherwise go to step 2.

Step 2: Compute

$$\Delta^\beta = \sum_{j\varepsilon J_+^\beta} (x_j^\beta - ub_j) \text{ where } J_+^\beta = \{j\varepsilon J^\beta : x_j^\beta \geq ub_j\}$$

and

$$\nabla^\beta = \sum_{j\varepsilon J_-^\beta} (\ell b_j - x_j^\beta) \text{ where } J_-^\beta = \{j\varepsilon J^\beta : x_j^\beta \leq \ell b_j\}$$

Step 3: If $\Delta^\beta \geq \nabla^\beta$ define $x_j^* = ub_j \; j\varepsilon J_+^\beta$ and let

$$J^{\beta+1} = J^\beta - J_+^\beta, \; P^{\beta+1} = P^\beta - \sum_{j\varepsilon J_+^\beta} ub_j .$$

If $\Delta^\beta < \nabla^\beta$ define $x_j^* = \ell b_j \; j\varepsilon J_-^\beta$ and let

$$J^{\beta+1} = J^\beta - J_-^\beta, \; P^{\beta+1} = P^\beta - \sum_{j\varepsilon J_-^\beta} \ell b_j .$$

If $J^{\beta+1} = \phi$ stop. The solution $x_j^* \; j\varepsilon J^\circ$ generated by the algorithm is optimal. Otherwise let β = β+1 and go to step 1.

Since at each iteration the set J^β is reduced by at least one element, the algorithm is finite. The algorithm relies on the fact that it is generally much easier to solve N(β) than (N). Problem N(β) can be solved by using its Kuhn-Tucker conditions. In fact, in several practical applications it is even possible to obtain a solution in closed form.

To prove that x_j^* $j \varepsilon J^\circ$, generated by the algorithm, solve (N), we construct a corresponding Kuhn-Tucker vector through the following results. Let λ^β be the Kuhn-Tucker multiplier associated to the knapsack constraint in $N(\beta)$.

<u>Lemma 1</u>: If at iteration β $\Delta^\beta \geq \nabla^\beta$ then

a) for any $s\varepsilon J_+^\beta$ we have $-DG_s(ub_s) \geq -DG_i(ub_i)$ for all $i\varepsilon J^\beta - J_+^\beta$ and $-DG_s(ub_s) \geq \lambda^\beta$

b) $\lambda^{\beta+1} \leq \lambda^\beta$

<u>Proof</u>:

a) $\lambda^\beta = -DG_j(x_j^\beta)$ $j\varepsilon J^\beta$. Let $s\varepsilon J_+^\beta$ and $i\varepsilon J^\beta - J_+^\beta$. Since the functions $G_j(\cdot)$ are convex, for any pair x_j^2, x_j^1 in X_j we have [10]:

$$[DG_j(x_j^2) - DG_j(x_j^1)](x_j^2 - x_j^1) \geq 0 \tag{10}$$

It follows that

$$-DG_s(ub_s) \geq -DG_s(x_s^\beta) = \lambda^\beta = -DG_i(x_i^\beta) \geq -DG_i(ub_i)$$

b) $$\sum_{j\varepsilon J^\beta - J_+^\beta} x_j^\beta = P^\beta - \sum_{j\varepsilon J_+^\beta} x_j^\beta \leq P^\beta - \sum_{j\varepsilon J_+^\beta} ub_j = P^{\beta+1} = \sum_{j\varepsilon J^{\beta+1}} x_j^{\beta+1}$$

For at least one $j_o\varepsilon J^{\beta+1}$ we have $x_{j_o}^{\beta+1} \geq x_{j_o}^\beta$. Thus

$$\lambda^{\beta+1} = -DG_{j_o}(x_{j_o}^{\beta+1}) \leq -DG_{j_o}(x_{j_o}^\beta) = \lambda^\beta$$

where the inequality follows from (10). $\blacksquare$

Similarly,

<u>Lemma 2</u>: If at iteration β $\Delta^\beta < \nabla^\beta$ then

a) for any $s\varepsilon J_-^\beta$ we have $-DG_s(\ell b_s) \leq -DG_i(\ell b_i)$ for all $i\varepsilon J^\beta - J_-^\beta$ and $-DG_s(\ell b_s) \leq \lambda^\beta$

b) $\lambda^{\beta+1} \geq \lambda^\beta$

The proofs of this lemma and of theorem 4 are omitted because they are similar to those of lemma 2 and theorem 3 respectively.

<u>Theorem 3</u>: Assume that $\Delta^{\beta} \geq \nabla^{\beta}$, $\Delta^{i} < \nabla^{i}$ $i=\beta+1,\beta+2,\ldots,\gamma-1$. Then

a) $J^{\gamma} \supseteq (J^{\beta}-J_{+}^{\beta}-J_{-}^{\beta})$

b) $\lambda^{\beta} \geq \lambda^{\gamma}$

<u>Proof</u>:

$$p^{\beta+1} = \sum_{j\epsilon J^{\beta+1}} x_j^{\beta+1} = \sum_{j\epsilon J^{\beta}-J_{+}^{\beta}} x_j^{\beta} - \sum_{j\epsilon J_{+}^{\beta}} ub_j + \sum_{j\epsilon J_{+}^{\beta}} x_j^{\beta} = \sum_{j\epsilon J^{\beta+1}} x_j^{\beta}+\Delta^{\beta} \tag{11}$$

$$p^{\beta+2} = \sum_{j\epsilon J^{\beta+2}} x_j^{\beta+2} = \sum_{j\epsilon J^{\beta+1}} x_j^{\beta+1} - \sum_{j\epsilon J_{-}^{\beta+1}} \ell b_j =$$

$$= \sum_{j\epsilon J^{\beta+1}-J_{-}^{\beta+1}} x_j^{\beta} + \sum_{j\epsilon J_{-}^{\beta+1}} x_j^{\beta} + \Delta^{\beta} - \sum_{j\epsilon J_{-}^{\beta+1}} \ell b_j .$$

Thus, since $J^{\beta+2} = J^{\beta+1}-J_{-}^{\beta+1}$

$$p^{\beta+2} = \sum_{j\epsilon J^{\beta+2}} x_j^{\beta} + \Delta^{\beta} - \sum_{j\epsilon J_{-}^{\beta+1}} (\ell b_j - x_j^{\beta}). \tag{12}$$

Similarly we obtain

$$p^{\gamma} = \sum_{j\epsilon J^{\gamma}} x_j^{\beta} + \Delta^{\beta} - \sum_{s=\beta+1}^{\gamma-1} \sum_{j\epsilon J_{-}^{s}} (\ell b_j - x_j^{\beta}). \tag{13}$$

From (11) it follows that for at least one $j_o \epsilon J^{\beta+1}$

$$x_{j_o}^{\beta+1} \geq x_{j_o}^{\beta} \quad . \tag{14}$$

But, from the Kuhn-Tucker conditions for $N(\beta)$ and $N(\beta+1)$:

$$-DG_i(x_i^{\beta}) = \lambda^{\beta} \text{ and } -DG_i(x_i^{\beta+1}) = \lambda^{\beta+1} \quad i\epsilon J^{\beta+1} \tag{15}$$

Combining (10), (14), (15), and lemma 1b) it follows that

$$x_j^{\beta+1} \geq x_j^{\beta} \quad \text{for all } j\epsilon J^{\beta+1}. \tag{16}$$

Note that if $\Delta^\beta = 0$, then $x_j^{\beta+1} = x_j^\beta \; j\varepsilon J^{\beta+1}$ solve $N(\beta+1)$ and if $\Delta^\beta > 0$, at least one inequality in (16) will be a strict inequality. Moreover, if the functions G_j are not all strictly convex, it is possible that $-DG_{j_o}(x_{j_o}^{\beta+1}) = \lambda^{\beta+1} = \lambda^\beta$. In this case $N(\beta+1)$ may have more than one optimal solution. However, at least one will satisfy (16). Thus

$$J_-^{\beta+1} \subseteq J_-^\beta \quad \text{and} \quad \Delta^\beta - \sum_{j\varepsilon J_-^{\beta+1}} (\ell b_j - x_j^\beta) \geq 0. \tag{17}$$

Similarly, (12), (17), and the fact that $J^{\beta+2} = J^{\beta+1} - J_-^{\beta+1}$ imply that

$$x_j^{\beta+2} \geq x_j^\beta \text{ for all } j\varepsilon J^{\beta+2},\; J_-^{\beta+2} \subseteq J_-^\beta \text{ and}$$

$$\Delta^\beta - \sum_{j\varepsilon J_-^{\beta+1}} (\ell b_j - x_j^\beta) - \sum_{j\varepsilon J_-^{\beta+2}} (\ell b_j - x_j^\beta) \geq 0.$$

Thus,

$$P^{\beta+3} = \sum_{j\varepsilon J^{\beta+3}} x_j^{\beta+3} \geq \sum_{j\varepsilon J^{\beta+3}} x_j^\beta \quad .$$

Continuing with the same reasoning, we obtain

$$J_-^i \subseteq J_-^\beta \quad i=\beta+1,\ldots,\gamma-1; \quad \Delta^\beta - \sum_{s=\beta+1}^{\gamma-1} \sum_{j\varepsilon J_-^s} (\ell b_j - x_j^\beta) \geq 0$$

and from (13)

$$P^\gamma = \sum_{j\varepsilon J^\gamma} x_j^\gamma \geq \sum_{j\varepsilon J^\gamma} x_j^\beta. \tag{18}$$

These conclusions together with $J^{\beta+1} = J^\beta - J_+^\beta$ and $J^{i+1} = J^i - J_i^i$ $i=\beta+1,\ldots,\gamma-1$ prove part a). From (18) it follows that $x_j^\gamma \geq x_j^\beta \; j\varepsilon J^\gamma$. Thus, from (10) and (15) with γ instead of $\beta+1$

$$\lambda^\gamma = -DG_j(x_j^\gamma) \leq -DG_j(x_j^\beta) = \lambda^\beta \quad .$$

<u>Theorem 4</u>: Assume that $\Delta^\beta < \nabla^\beta$, $\Delta^i \geq \nabla^i$ $i=\beta+1,\beta+2,\ldots,\gamma-1$. Then

a) $J^\gamma \supseteq (J^\beta - J^\beta_- - J^\beta_+)$

b) $\lambda^\beta \leq \lambda^\gamma$

<u>Theorem 5</u>: The set x^*_j $j\varepsilon J°$ generated by the algorithm is optimal in (N).

<u>Proof</u>: By lemmas 1 and 2, theorems 3b) and 4b), the set x^*_j $j\varepsilon J°$ generated by the algorithm has the following property:

$$-DG_{k_1}(ub_{k_1}) \geq \ldots \geq -DG_{k_p}(ub_{k_p}) \geq -DG_{v_1}(x^*_{v_1}) = \ldots$$

$$= -DG_{v_g}(x^*_{v_g}) \geq -DG_{i_1}(\ell b_{i_1}) \geq \ldots \geq -DG_{i_s}(\ell b_{i_s}) \qquad (19)$$

where, the indices $k_1, k_2,\ldots, k_p$; $i_1, i_2,\ldots, i_s$, and $v_1, v_2,\ldots, v_g$ correspond to variables for which the optimal value was set at the upper bound and lower bound in step 3 of the algorithm and variables whose optimal value was obtained in step 1 of the last iteration of the algorithm respectively.

To see that conditions (3) - (9) are satisfied take,

$$\lambda = -DG_{v_1}(x^*_{v_1}),$$

$$\tau_{i_j} = \lambda + DG_{i_j}(\ell b_{i_j}) \geq 0, \quad u_{i_j} = 0 \quad j=1,2,\ldots,s$$

$$u_{k_j} = -DG_{k_j}(ub_{k_j}) - \lambda \geq 0, \quad \tau_{k_j} = 0 \quad j=1,2,\ldots,p \text{ and}$$

$$\tau_{v_j} = u_{v_j} = 0 \quad j=1,2,\ldots,g.$$

Thus, since x^*_j $j\varepsilon J°$ also satisfies (6), (7), and (9) it follows that

$$x_{k_j} = ub_{k_j} \quad j=1,2,\ldots,p; \quad x_{v_j} = x^*_{v_j} \quad j=1,2,\ldots,g; \quad \text{and}$$

$x_{i_j} = \ell b_{i_j}$ $j=1,2,\ldots,s$ solve (N). $\blacksquare$

Since the algorithm depends strongly on the existence of a solution to $N(\beta)$ we prove the following theorem. Let $x_j(\lambda)$ denote a point in R where $DG_j[x_j(\lambda)] = \lambda$.

Theorem 6: If $G_j(\cdot)$ $j\varepsilon J^\beta$ are strictly convex and differentiable on X_j then, a necessary and sufficient condition for $N(\beta)$ to have a solution is that there exist λ_1 and λ_2 such that

$$\sum_{j\varepsilon J^\circ} x_j(\lambda_1) \leq P^\beta \leq \sum_{j\varepsilon J^\circ} x_j(\lambda_2) \text{ with}$$

$$x_j(\lambda_1)\varepsilon X_j \text{ and } x_j(\lambda_2)\varepsilon X_j \quad j\varepsilon J^\beta$$

Proof: Sufficient Condition: The functions $x_j(\lambda)$ $j\varepsilon J^\beta$ are continuous. Thus, $\sum_{j\varepsilon J^\beta} x_j(\lambda)$ is a continuous function of λ. Moreover, X_j $j\varepsilon J^\beta$ are open intervals in R. Therefore, there are a $\lambda_o\varepsilon R$ and points $x_j(\lambda_o)\varepsilon X_j$ $j\varepsilon J^\beta$ such that $\sum_{j\varepsilon J^\beta} x_j(\lambda_o) = P^\beta$.

Necessary Condition: Follows from the Kuhn-Tucker conditions for $N(\beta)$. ■

It is worth pointing out that a necessary condition for $N(\beta)$ to have a solution is that

$$\max_{j\varepsilon J^\beta} \lim_{\substack{x_j\to a_j \\ x_j\varepsilon X_j}} DG_j(x_j) < \min_{j\varepsilon J^\beta} \lim_{\substack{x_j\to b_j \\ x_j\varepsilon X_j}} DG_j(x_j) \qquad (20)$$

where

$a_j = \inf\{x\varepsilon R\colon x_j\varepsilon X_j\}$ $j\varepsilon J^\beta$ and

$b_j = \sup\{x\varepsilon R\colon x_j\varepsilon X_j\}$ $j\varepsilon J^\beta$

Condition (20) can be useful as a tool to conclude that $N(\beta)$ has no solution.

When there are strictly increasing and decreasing functions over X_j among the functions $G_j(\cdot)$ $j\varepsilon J^\beta$, $N(\beta)$ has no solution. In this case the left hand side of inequality (20) is negative. The next theorem shows how to cope with this difficulty.

Let $J_1 = \{j \varepsilon J^\circ\colon G_j(\cdot) \text{ is strictly increasing}\}$, $J_1 \neq \emptyset$ and let $J_2 = \{j \varepsilon J^\circ\colon G_j(\cdot) \text{ is strictly decreasing}\}$, $J_2 \neq \emptyset$. Note that we do not require $J^\circ - J_1 - J_2$ to be an empty set. Assume that (N) has an optimal solution x_j^* $j\varepsilon J^\circ$, $ub_j < +\infty$ and $\ell b_j > -\infty$ $j \varepsilon J^\circ$.

Theorem 7: The optimal solution of (N) is such that either

$$x_i^* = ub_i \quad i\varepsilon J_2 \text{ and/or } x_i^* = \ell b_i \quad i \varepsilon J_1.$$

Proof: The optimal solution satisfies (3) - (9) and in particular

$$-DG_j(ub_j) = \lambda + u_j \quad j\varepsilon J(ub) \equiv \{j\varepsilon J^\circ\colon x_j^* = ub_j\}$$

$$-DG_j(\ell b_j) = \lambda - \tau_j \quad j\varepsilon J(\ell b) = \{j\varepsilon J^\circ\colon x_j^* = \ell b_j\} \text{ and}$$

$$-DG_j(x_j^*) = \lambda \quad j\varepsilon J^\circ - J(\ell b) - J(ub).$$

Note that the assumption $P^\circ > \sum_{j\varepsilon J^\circ} \ell b_j$ implies $J^\circ - J(\ell b) \neq \emptyset$. If $\min\{-DG_j(x_j^*) \; j\varepsilon J^\circ - J(\ell b)\} \geq 0$, since $-DG_i(x_i) < 0$ $i\varepsilon J_1$, it follows that $J_1 \subseteq J(\ell b)$. Otherwise, since $-DG_i(x_i) > 0$ $i\varepsilon J_2$ and $-DG_j(ub_j) \geq -DG_i(x_i^*) \geq -DG_k(\ell b_k)$ for any $j\varepsilon J(ub)$ $i\varepsilon J^\circ - J(ub) - J(\ell b)$ and $k\varepsilon J(\ell b)$, we have that $J_2 \subseteq J(ub)$. $\blacksquare$

A direct consequence of theorem 7 is that the solution to (N) can be obtained by solving the following two problems

$$(N_1)\colon \quad z_1 = \min\{\sum_{j\varepsilon J^\circ - J_2} G_j(x_j)\colon \sum_{j\varepsilon J^\circ - J_2} x_j = P^\circ - \sum_{j\varepsilon J_2} ub_j \;\; \ell b_j \leq x_j \leq ub_j \;\; j\varepsilon J^\circ - J_2\}$$

$$(N_2)\colon \quad z_2 = \min\{\sum_{j\varepsilon J^\circ - J_1} G_j(x_j)\colon \sum_{j\varepsilon J^\circ - J_1} x_j = P^\circ - \sum_{j\varepsilon J_1} \ell b_j \;\; \ell b_j \leq x_j \leq ub_j \;\; j\varepsilon J^\circ - J_1\}$$

and taking $z = \min\left(\sum_{j\varepsilon J_2} G_j(ub_j) + z_1, \sum_{j\varepsilon J_1} G_j(\ell b_j) + z_2\right)$. By analyzing the Kuhn-Tucker conditions (3) - (9) it becomes apparent that problem (N) has no solution if there are simultaneously a strictly increasing $G_{j_1}(\cdot)$ with $\ell b_{j_1} = -\infty$ and a strictly decreasing $G_{j_2}(\cdot)$ with $ub_{j_2} = +\infty$ and these two limits are attainable on X_{j_1} and X_{j_2} respectively. Our algorithm does not apply in these instances. However, it is not difficult to show that Theorem 7 holds if only decreasing functions (increasing functions) have unbounded upper bounds (lower bounds).

Concluding this section we consider a version of problem (N) when the knapsack constraint is an inequality. Let ($N\leq$) and ($N\geq$) be the versions of (N) with constraint 1 being an inequality of the type $\leq$ and $\geq$ respectively. For each index $j\varepsilon J°$ let h_j be the value of x_j for which $DG_j(x_j) = 0$ over X_j. If such point does not exist we adopt $h_j = -\infty$ ($+\infty$) in Theorem 8 (Theorem 9) below. Let x_j^* $j\varepsilon J°$ solve (N). The indices k_j, v_j, and i_j correspond to those in expression (19).

Theorem 8:

a) If $\lambda = -DG_{v_j}(x_{v_j}^*) \geq 0$, $x_j = x_j^*$ $j\varepsilon J°$ solve ($N\leq$)

b) If $= -DG_{v_j}(x_{v_j}^*) < 0$, x_j defined as:

$x_{i_j} = \ell b_{i_j}$ $j=1,2,\ldots,s$;

$x_{v_j} = \max(\ell b_{v_j}, h_{v_j})$ $j=1,2,\ldots,g$;

$x_{k_j} = \max(\ell b_{k_j}, h_{k_j})$ for all k_j such that $DG_{k_j}(ub_{k_j}) \geq 0$; and

$x_{k_j} = ub_{k_j}$ for all k_j such that $DG_{k_j}(ub_{k_j}) < 0$ solve ($N\leq$).

Theorem 9:

a) If $\lambda = -DG_{v_j}(x_{v_j}^*) \leq 0$, $x_j = x_j^*$ $j\varepsilon J°$ solve ($N\geq$)

b) If $= -DG_{v_j}(x_{v_j}^*) > 0$, x_j defined as:

$x_{k_j} = ub_{k_j}$ $j=1,2,\ldots,p$;

$x_{v_j} = \min(ub_{v_j}, h_{v_j})$ $j=1,2,\ldots,g$;

$x_{i_j} = \min(ub_{i_j}, h_{i_j})$ for all i_j such that $DG_{i_j}(\ell b_{i_j}) \leq 0$; and

$x_{i_j} = \ell b_{i_j}$ for all i_j such that $DG_{i_j}(\ell b_{i_j}) > 0$ solve ($N\geq$).

Theorems 8 and 9 show that problems ($N\leq$) and ($N\geq$) can be solved by first applying our algorithm to solve (N) and next making appropriate changes to x_j^* $j \varepsilon J$. The proofs of both theorems have been omitted because of their simplicity. The extensions of the last two results for the case where the solution to problem (N) has all variables at the upper or lower bound is straightforward.

3. COMPUTATIONAL RESULTS

Tables 1 through 7 present the results of solving 84 problems of type (N). All problems were solved by the algorithm described in this paper and whenever applicable by Luss and Gupta's method [9]. For identification purposes, our algorithm is denoted by BH, while the algorithm in [9] is denoted by LG. In each problem, the objective function was composed by functions $G_i(x_i)$ of the same functional form. They are indicated in the first row of each table. Following the time in seconds to solve each problem, by both methods, is the number of iterations required. n represents the number of variables in a problem. For a fixed n we solved three problems for each type of objective function. In Luss and Gupta's method [9] the ordering of the derivatives evaluated at the lower bound of each variable was executed by the "Quicksort Method" [7]. In our algorithm, problems $N(\beta)$ were solved using the corresponding Kuhn-Tucker conditions. Luss and Gupta's algorithm does not apply to the problems of Tables 6 and 7, because either the objective functions are strictly convex increasing (Table 6) or we have not imposed, as their algorithm requires, any condition among the values of the bounds, $P°$ and the point where each of the $G_i(x_i)$ attains its minimum (Table 7). The computer used is a Borroughs B6700. The programs were written in Algol. Applications of problems presented in Tables 5 and 7 to hierarchical production planning can be found in [2] and [3]. The parameters ($s_i, m_i, \ell b_i, ub_i$, etc.) corresponding to problems of Tables 1 through 5 (6 and 7) were randomly generated in intervals where the functions $G_i(x_i)$ are strictly convex decreasing (strictly convex).

For the problems presented in Tables 1 through 5 we have noticed that the time required by Luss and Gupta's method just to compute the derivatives at the lower bound and order them is comparable to the total time of our algorithm.

As a final note, we would like to point out that, although we have tried to program both algorithms as efficiently as we could, the computational results should be looked upon with caution. The intent of including the results of these experiments is to give to the reader an idea of the time required to solve problems of the type (N) with different objective functions and sizes. The major attraction of our algorithm is its applicability to a wider class of problems than the one in [9].

ACKNOWLEDGEMENT

The authors would like to thank Mr. Luis Antonio Malheiros Meloni for his cooperation in the programming of the algorithms and to Thomas L. Magnanti and Debashish Sarkar for their careful reading of an earlier version of the paper.

Table 1

$G_i(x_i) = s_i[1-\exp(-m_i x_i)]$ $i=1,2,\ldots,n$ $m_i > 0$

	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.138	0.157	0.119	0.292	0.213	0.237	0.482	0.540	0.434	0.709	0.720	0.711
No. of Iterations	6	7	7	6	6	6	8	9	7	8	8	9
LG (sec)	0.728	1.029	1.174	1.125	4.179	0.830	5.903	9.073	3.969	10.531	2.109	15.124
No. of Iterations	2	4	8	1	7	1	3	4	2	3	1	5

Table 2

$G_i(x_i) = s_i \ln[1+m_i x_i]$ $i=1,2,\ldots,n$ $m_i > 0$

	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.051	0.050	0.038	0.118	0.099	0.131	0.163	0.178	0.162	0.191	0.245	0.208
No. of Iterations	4	4	3	6	5	6	6	6	4	7	6	5
LG (sec)	0.486	0.447	0.201	0.516	1.316	0.488	0.536	0.958	1.232	3.394	4.210	1.105
No. of Iterations	3	3	1	2	4	2	1	2	1	6	4	1

Table 3

	$G_i(x_i) = s_i \frac{x_i+c_i}{x_i+m_i}$ with $(m_i > c_i)$ $i=1,2,\ldots,n$											
	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.113	0.095	0.124	0.208	0.221	0.178	0.308	0.307	0.298	0.400	0.421	0.327
No. of Iterations	6	5	5	5	5	5	4	6	4	5	6	3
LG (sec)	0.210	0.284	0.229	0.825	0.796	0.291	1.033	2.687	1.098	1.394	5.448	2.543
No. of Iterations	1	1	1	3	2	1	1	4	1	1	2	1

Table 4

	$G_i(x_i) = s_i x_i - m_i x_i^2$ with $\ell b_i < k_i$ and $P < \sum_{i=1}^{n} \min(ub_i, k_i)$ where $k_i = \frac{s_i}{2m_i}$											
	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.063	0.059	0.049	0.105	0.108	0.101	0.152	0.161	0.157	0.223	0.261	0.210
No. of Iterations	7	7	6	8	7	6	8	8	8	8	8	8
LG (sec)	0.225	0.236	0.164	0.361	0.850	0.283	0.737	1.129	0.693	2.150	0.893	0.745
No. of Iterations	16	12	1	5	34	1	10	13	10	18	1	7

Table 5

	$G_i(x_i) = a_i/x_i \quad i=1,2,\ldots,n$											
	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.061	0.058	0.048	0.095	0.076	0.133	0.128	0.246	0.168	0.276	0.306	0.282
No. of Iterations	6	3	4	5	6	7	3	6	5	8	4	3
LG (sec)	0.624	0.752	0.916	2.611	2.012	1.431	2.618	2.715	4.869	3.719	3.901	5.088
No. of Iterations	2	1	3	4	2	1	2	5	5	6	2	1

Table 6

	$G_i(x_i) = \exp k_i x_i \quad k_i > 0 \quad i=1,2,\ldots,n$											
	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.163	0.151	0.135	0.237	0.267	0.261	0.445	0.443	0.381	0.596	0.486	0.536
No. of Iterations	7	7	6	5	6	6	7	7	7	8	6	6

Table 7

	$G_i(x_i) = \frac{1}{2}(\frac{h}{S} - \frac{x_i}{S_i})^2 \quad i=1,2,\ldots,n$											
	n = 50			n = 100			n = 150			n = 200		
BH (sec)	0.054	0.050	0.044	0.114	0.073	0.073	0.159	0.151	0.170	0.202	0.214	0.227
No. of Iterations	6	6	5	6	3	3	5	4	5	4	5	8

REFERENCES

1. R. D. Armstrong, W. D. Cook, and F. E. Palacios-Gomez, "An Algorithm for a Nonlinear Discontinuous Knapsack Problem", Management Science 25, 884-894 (1980).

2. G. R. Bitran and A. C. Hax, "On the Design of Hierarchical Production Planning Systems", Decision Sciences 8, 28-55 (1977).

3. G. R. Bitran, E. Haas, and A. C. Hax, "Hierarchical Production Planning, Part I", Operations Research Center, M.I.T., Technical Report, 1980.

4. A. Charnes and W. W. Cooper, "The Theory of Search: Optimum Distribution of Search Effort", Management Science 5, 44-49 (1958).

5. M. Held, P. Wolfe, and H. P. Crowder, "Validation of Subgradient Optimization", Mathematical Programming 6, 62-88 (1974).

6. C. C. Holt, F. Modigliani, J. F. Muth, and H. A. Simon, Planning Production, Inventories, and Work Force, Prentice Hall, Englewood Cliffs, NJ, 1960.

7. D. E. Knuth, The Art of Computer Programming, Addison Wesley, Reading, MA, 1973, Volume 3.

8. S. Love, Inventory Control, McGraw-Hill, New York, 1979.

9. H. Luss and S. K. Gupta, "Allocation of Effort Resources Among Competing Activities", Operations Research 23, 360-366 (1975).

10. O. L. Mangasarian, Nonlinear Programming, McGraw-Hill, New York, 1969.

11. C. Wilkinson and S. K. Gupta, "Allocating Promotional Effort to Competing Activities: A Dynamic Programming Approach", IFORS Conference, Venice, 419-432 (1969).